

Association for Information Systems AIS Electronic Library (AISeL)

ICIS 1989 Proceedings

International Conference on Information Systems
(ICIS)

1989

DIVISION AMONG THE RANKS: THE SOCIAL IMPLICATIONS OF CASE TOOLS FOR SYSTEM DEVELOPERS

Wanda J. Orlikowski
Massachusetts Institute of Technology

Follow this and additional works at: <http://aisel.aisnet.org/icis1989>

Recommended Citation

Orlikowski, Wanda J., "DIVISION AMONG THE RANKS: THE SOCIAL IMPLICATIONS OF CASE TOOLS FOR SYSTEM DEVELOPERS" (1989). *ICIS 1989 Proceedings*. 40.
<http://aisel.aisnet.org/icis1989/40>

This material is brought to you by the International Conference on Information Systems (ICIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in ICIS 1989 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.

DIVISION AMONG THE RANKS: THE SOCIAL IMPLICATIONS OF CASE TOOLS FOR SYSTEM DEVELOPERS

Wanda J. Orlikowski
Sloan School of Management
Massachusetts Institute of Technology

ABSTRACT

This paper explores how the introduction of CASE tools in systems development changes the social relations among project team members. An investigation into the role of CASE tools on projects found structural changes due to modification of the systems development division of labor and shifts in patterns of dependency among project team coalitions. These changes triggered a polarization among the system developers which was evinced in acts of coercion and rebellion, the display of territorialism, resentment, and stereotyping, as well as the enactment of subcultures. These findings are interpreted within a broader social theoretic framework, and their implications for research and practice are discussed.

1. INTRODUCTION

Today we are seeing tremendous interest and investment in the automation of the systems development process. This trend towards Computer-Aided Software Engineering (CASE) tools is an attempt to remedy the apparent lack of computer-based support for systems developers, a lack to which many of the ills of the systems building process have been attributed. CASE tools are software programs which automate or support tasks typically constituting information systems development practice. There is no general agreement as to what functionality a CASE tool should provide, but most would agree that CASE tools comprise some subset of the following elements: screen and report design aids, text and diagram editors, data modeling tools, data dictionaries, code generators, testing and debugging tools. The major advantages that have been advertised for CASE tools include increased responsiveness to user needs in the face of changing requirements, increased systems development productivity, decreased systems development time, enhanced system quality, standardization, ability to replace project personnel easily, and the capability to solve larger and more complex problems (Case 1985; Freedman 1986; Stamps 1987).

While some of these benefits may be realizable, the mechanisms through which CASE tools are successfully implemented are not identified. There are few, if any, detailed analyses of actual CASE tool implementations, and little empirical data is available on the organizational effects of using automated means to develop systems. Most discussions of CASE tools focus on technical and project management criteria with little discussion, and hence little understanding, of the social implications of using CASE tools. Information technology, as is by now well accepted, can never be deployed in a vacuum. Its form and function are always influenced by the social context within which it is embedded (Boland 1979; Kling

and Scacchi 1982; Markus 1983; Weick 1984) and it invariably exerts a reciprocal influence on that context (Giddens 1984). Similarly, we can expect that the information technology deployed to support/automate systems development (CASE tools) will interact with the organizational context, introducing perturbations into the social relations surrounding tool development and use. Drawing on an empirical study, this paper recounts how the deployment of CASE tools in systems development changed social relations among developers, providing insight into some organizational changes triggered by automating systems development.

The following section provides background to the research study, outlining the organizational context and history within which CASE tools were introduced. Section three investigates the structural changes engendered by CASE tools and describes the behavioral response of system developers. Section four discusses the meanings of these changes for the developers in terms of their perspectives and subculture affiliations. Section five reviews the findings, recasting them in terms of a more general, social theory. The paper concludes by outlining some implications for practice and for future research into the social issues surrounding CASE tools.

2. BACKGROUND TO THE RESEARCH

2.1 The Research Study

The discussion in this paper draws on a research study that investigated the automation of the systems development process in a large, northeastern software consulting firm (henceforth known as the Beta Consulting Corporation). The software consulting firm, in operation since the 1960s, employs about 600 consultants and develops computer-based information systems for its clients across various

industries: financial services, manufacturing, retail, and government. These information systems are typically large, transaction-processing applications used by clients to support their major administrative activities. Beta's operations are organized by project, with project teams varying from around ten to over a hundred personnel, and projects extending from a few months to a number of years in duration. Project costs range from a hundred thousand to a few million dollars. As a consequence of the growing demand for large, complex, integrated application software, Beta has -- over the last two decades -- attempted to streamline as much of its systems development practice as possible. The most recent and visible manifestation of this strategy is the construction and deployment of CASE tools within project teams. This shift towards computer-based systems development support in Beta dates back more than five years, in contrast to most firms which have not yet seriously committed to CASE.¹

The findings discussed here are part of a larger research study that focused on the organizational changes that accompanied Beta's implementation of CASE tools in its systems development operations (Orlikowski 1988). The study employed ethnographic techniques (Agar 1980; Van Maanen 1979, 1988) such as observation of participants, interaction with CASE tools, documentation review, social contact, unstructured and semi-structured interviews. It was executed over eight months within Beta and in those client sites where Beta developers were building application systems. In the first phase of the research, historical data on the Beta corporation and its systems development practices was gathered from published material (in-house and trade press), and from interviews with senior managers who had been involved in Beta's traditional systems development, as well as its adoption of a computer-based systems development process.

With some background information on Beta and its practices, five different application projects (four large and one small) were selected for indepth analyses. Projects were not selected at random but were strategically identified to guarantee exposure to the use of CASE tools in all major phases of the systems development life cycle (requirements analysis, conceptual design, detailed design, implementation, and testing). An average of four weeks was spent on each project, observing and interviewing team members in their daily systems development work, and in their interaction with each other and the CASE tools. There were 120 interviews conducted, each lasting an average of one and a half hours. Participation in the research was voluntary and while the particular projects examined were selected by Beta's senior management, individuals within projects were invited to participate in the study by the researcher alone. These individuals spanned Beta's hierarchic levels from the most junior analysts and programmers to senior project managers.² Other key informants, such as the senior recruiting officer, the director of research and development, sales directors, major client managers, and former Beta employees, were

identified and sought out both within and outside Beta. Data was also collected throughout the study at monthly (all day) division meetings and in project training sessions on CASE tools.

Beta has not diffused its CASE tools on a corporate-wide basis, but has followed a phased implementation with major offices first adopting the tools, followed by the smaller offices. Because offices regularly share personnel to take advantage of slack resources, it was possible to find developers having differential exposure to the CASE tools. Thus, on the projects studied, there were developers who had never used CASE tools before, as well as developers who were four and five year veteran tool users. This differential exposure to tool use facilitated a natural contrast that revealed interesting insights into how systems developers perceive and interact with CASE tools.

2.2 Project Teams Before the Use of CASE Tools

The history of project team composition in Beta indicates that gradually over the last two decades a partitioning between functional and technical expertise was established within system development teams.³ This partitioning, however, was only institutionalized with the introduction of CASE tools during the last five years, when a number of structural changes underscored the division among the project team members, accentuating relations of power and dependence. Before proceeding to the discussion of these organizational changes, the evolution of project team relations in Beta over time is sketched out.

In the late 1960s, Beta personnel working on systems development projects were not differentiated by expertise so much as by what stage in the systems development process they serviced. Thus, some people specialized in the upfront conceptual work of systems specification, performing information requirements determination and systems analysis, while others conducted actual system implementations via functional and technical design, programming, testing, and installation. While this type of specialization did lead to differentiated knowledge and experience, the lines were not drawn along functional and technical expertise. Largely as a result of the unanticipated weaknesses of this temporal schism -- having two different teams develop a single system for a client often resulted in many development inefficiencies⁴ as well as duplicate client negotiations -- the tasks of the development process were bundled together, so that a single team carried a project through in its entirety.

A division of labor which encouraged functional and technical specialization quickly emerged within such a single project team structure. This specialization was driven by the size and complexity of applications that Beta began to develop and was encouraged by the software engineering tenets gaining ascendancy in the early 1970s.

Even as different tasks were assigned to various team-appointed specialists, all the tasks concerned the building of an application -- the target system -- for the current client. None of the project team members were responsible for supporting the activities of other team members. Hence, dependencies on the project resulted from coordinating disparate tasks, rather than from a differential distribution of resources. Functional and technical specializations on teams were informally negotiated and sustained, were temporary (lasting only the duration of a project), and were not officially recognized by Beta's structural apparatus (its hierarchy, assignment mechanism, promotion and reward schedule). In the early 1970s, Beta did not formally differentiate its personnel along lines of expertise.

3. RESPONDING TO CASE TOOLS AND STRUCTURAL CHANGE

3.1 Setting the Stage for CASE Tools

In the mid-1970s, however, information technology became increasingly integrated, diverse and technically complex, encompassing different hardware and software standards, sophisticated operating systems, database management systems, networks, and multiple computer configurations. Beta recognized that in order to deliver "leading edge" systems, project teams would have to augment their level of technical expertise. Senior Beta management formally designated some individuals as the firm's technical "experts" and a separate division within Beta was formed to house them. Personnel for the technical division were specifically recruited from computer science schools and were not assigned to particular project teams. Instead of spending time on clients' sites building application systems, they were located at Beta's headquarters to research new technological innovations and to serve as general technical consultants to projects on an as-needed basis. This latter responsibility required them to travel to projects for a few days at a time to provide the technical knowledge that local project teams lacked. While the technical expertise of these specialists was formally recognized by Beta, little power accrued to them as their involvement with projects was minor, and it is project engagements that earn revenues and hence influence within Beta. The technical specialists were generally perceived as advisors and referred to as the "consultants' consultants." The technical expertise furnished was strictly concerned with the hardware and software environment within which application systems were being built. There was, as yet, no notion of using information technology to support systems development work.

Beta's systems development practice grew and the demand for technically complex application systems increased. Time pressure on technical specialists limited even further their already sporadic contact with project teams and, as a result, their contributions to local systems development efforts were shortlived and superficial. The general

sentiment from local project managers was that these technical specialists were too inexperienced in functional matters, too focused on narrow technologies, and too removed from the daily exigencies of projects to benefit applications development. Out of these pressures and frustrations, the concept of "localized technical groups" emerged in Beta about ten years ago. Each local office now develops its own cadre of technical experts -- consultants specializing in technical matters -- to support client projects specific to each local office through close and daily involvement in systems development.

By establishing local technical groups, Beta personnel in local offices are formally differentiated by functional or technical expertise. Like their functional counterparts, technical consultants are assigned to client projects on a full-time basis. Each project team now comprises two distinct types of personnel: the "functional team" (drawn from the general pool of local functional consultants), and the "technical team" (drawn from the local technical group). Initially, functional consultants developed application systems while technical consultants supported their functional colleagues in the technical aspects of application development. Technical consultants were the experts on CICS, VM/CMS, IMS, ADABAS, UNIX, telecommunications, performance tuning, the technical feasibility of various proposed designs, and so on. They were largely uninvolved in developing application systems, playing a staff role to the functional consultants' line role. In particular (with the rare exception of applications based on sophisticated telecommunications networks or esoteric systems software), they did not represent significant components on a project's critical path.

3.2 Enter CASE tools

With the growing sophistication of systems being demanded by clients, and the consequent increased system development time-frames and project costs, Beta management decided that local technical consultants should develop capabilities to support the project teams. At first, these capabilities comprised what came to be known as the technical architecture.⁵ Senior managers in Beta realized that they could leverage their projects by allowing a few of the technical consultants to build the generic, technical foundation for all of a project's application programs, rather than have each functional consultant develop their own. They found they could avoid duplicated effort and the inevitable inconsistencies that arise when team members work in isolation. In this way, time was saved and the work was easier to correct, validate, and refine (thus improving the quality of the product). There was also less need for functional consultants to be technically knowledgeable before they could develop application systems.

This represents somewhat of a shift in the work performed by technical consultants on projects, changing from a purely advisory role to one that builds a substantial portion

of the application system under development for the client. This technical architecture built by the technical team does not strictly constitute CASE tools, as it forms part of the client's application system rather than a technology owned by Beta. However, it clearly was its forerunner, as in practice it amounts to information technology that assists functional consultants' development work by eliminating their need to build complex technical procedures in their programs.

In the early 1980s, the idea of fully-fledged CASE tools, in-house technology dedicated to supporting systems developers, had taken hold in Beta and over time a number of different CASE tools were developed on separate Beta projects and quickly diffused throughout the firm. As CASE tools became more widely known throughout Beta, they began to be a common feature of all large system development projects, mediating many systems development practices. Concomitantly, the role of technical consultants on projects began to change. Not only were they responsible for providing technical advice and building the technical architecture, they were now also responsible for setting up, modifying, and maintaining Beta's CASE tools at each client site.⁶

3.3 Structural Changes Following CASE Tools

As the technical architecture and CASE tools became more critical to the systems development process, the technical team responsible for them began to play a pivotal role on each project. Reflecting this change in scope, the technical contingent on projects increased in size (from some five percent of the project team members to about twenty percent). The use of technical architectures and CASE tools radically changed the dependence relations of the project team. Whereas before, the functional consultants called on the technical team on an as-needed basis, now they had to rely on the technical consultants to set up the tools and the architecture before they could perform any substantive work. The technical team's installation of tools and building of the technical architecture had become key stages on the critical path of systems development projects. Further, the technical team had to become more intimately involved in analysis and design decisions. That is, the structure of the technical architecture and CASE tools were constraints on possible application designs, while the conceptual systems design influenced the content of the technical architecture and CASE tools implemented. The technical team began to assume more of a production role as opposed to its prior, purely support role, which had not involved any direct involvement in applications development work.

The division of labor on systems development projects had changed, with functional consultants relinquishing many technical tasks to the technical consultants, whose involvement in project activities had shifted from the periphery to the center. Accompanying the increased visibility and

activity of the technical team had come increased responsibility and power as the functional team became heavily dependent on the technical team, whose activities facilitated their productive work.

4. MEDIATING THE MEANING OF STRUCTURAL CHANGE

Following the increased dependence of project teams on a technical infrastructure (technical architecture and CASE tools), project social relations became polarized around two very different perspectives: the technical and the functional. Each of these perspectives represents a separate subculture that has formed within Beta, reflecting different perceptions of and interactions with clients, project goals, and systems development activities in general. The strength of the polarization was surprising given the fairly strong and relatively homogeneous corporate culture that pervades Beta, but it adds support to Riley's (1983) contention that multiple subcultures develop even within a more overarching culture. The following discussion examines how the norms, orientations, and interests of the different subcultures mediated individuals' understanding of and actions towards CASE tools and each other.

4.1 Functional Subculture

4.1.1 Functional Perspective

Functional consultants perform the substantive work of Beta's systems development practice. They do not perceive themselves as "system developers," but as "business consultants" who develop functional solutions for client business problems through the medium of information technology. The medium is considered less important than the functionality that is provided. Information technology is apprehended as the means through which valued ends are achieved, rather than an end in itself. Hence, the information technology used by functional consultants is valued for its instrumental contribution to their work. Their general stance towards human activity (Schein 1987, pp. 101-102) reflects a **results orientation**, that is, a focus on "doing" (achievement and accomplishment) rather than a focus on "being" (process and development). Their time orientation is the present and their context local. They concentrate their energies on building a system for the current situation and completing the immediate project. Their attitude towards information technology (both the CASE tools and the actual technology used to build client application systems) reflects their results orientation and instrumentalism; they objectify it. Thus, functional consultants, even though they use information technology as a raw material in their work -- building and fashioning a system for clients out of an amalgam of hardware and software -- appear to be unaware of, or to ignore, the often arbitrary and nontechnical aspects of information technology. They treat

CASE as a neutral, abstract object, that facilitates rational and deliberate manipulation and that can be deployed across client contexts, application domains, and problem types.

4.1.2 Conflict Over Expertise

When local technical groups were first initiated within Beta and technical consultants joined project teams to provide technical advice, functional consultants appreciated the support and service they received from these "backroom guys,"⁷ who resolved tricky technical issues in the systems being built, thereby helping functional consultants achieve their desired ends. The central role of the functional consultants as primary producers of application systems was not threatened or challenged. There was little felt dependence on the technical consultants, who kept a low profile and whose advice dealt with the information technology itself. The technical consultants did not in any substantial way shape the direction or content of work performed by functional consultants.

With the implementation of CASE tools on projects came a change in the responsibilities and role of technical consultants. Perceptions among functional consultants and relations between the technical and functional teams have been noticeably affected. The technical consultants are now seen to perform many development functions. They are no longer the "technical gurus" offering backstage advice; they have moved firmly into center stage, both in terms of the importance of their work to the project and in terms of the financial investment their work represents to Beta and its clients. The functional consultants feel somewhat upstaged and less in control of the exigencies of their work. The activities of the technical consultants are now on the critical path of the project, so they directly influence and constrain the work of functional consultants. Functional consultants are acutely aware of their dependence on the technical team, whose tools and technical architecture they are required to employ. A functional manager commented on the change.

The technical team had no formal role on the team before tools. They did technical advising and resolved technical issues. Now they are the key part of the project, building and running the engine that everyone works off.

There is resentment that the technical consultants no longer provide support to the functional consultants, who still regard themselves as the main actors. Technical consultants are perceived as having "stolen the show" and "doing their own thing," with little or no regard for the needs or activities of the functional players. A functional consultant expressed the sense of abandonment generally felt by the functional team.

Now (after CASE tools were introduced) there is a distinct lack of communication between the technical and functional teams, because we have mutually exclusive tasks and motivations. Before we worked on the same thing. Their focus was on providing technical support and expertise as needed and demanded by us, the functional team. They provided a service; they were the internal consultants. Now they produce a product, the program shells and the bridges, and they are less concerned with support. There has been a big change in roles between the two teams. The onus for solving technical problems is now a functional responsibility. The technical team feel they have other things to do than support us. But we feel they are there to support us....There is a concern that they are getting too caught up in the design, that they may dictate the design. It is important that the tech team assume an advisory role and not drive the design.

Such different perceptions indicate that functional and technical consultants have well-defined yet opposing expectations of each others' roles and responsibilities, resulting in conflict over expertise and the bounds of legitimate action. A subtle territorialism has emerged within Beta which is manifest in the stereotyping and subcultural activities engaged in by the groups of consultants.

4.1.3 Stereotyping the Technical Consultants

There is a deeply felt sense among the functional team that the technical consultants are just interested in the technology and in "hacking" the most elegant design or code without regard for what support the functional teams needs, or when. One functional consultant commented on the technical team's attitude by noting:

The technical team should rely more on the functional team than they do. They have their own ideas and don't want to know the functional story. They are not open to criticism. They feel some ownership of the tools and so are very defensive. I guess that's human nature...they just don't want to know that their tools are defective or weak.

A lot of "finger pointing" is apparent, with each team using the other as a scapegoat when things go wrong. Typical comments from functional consultants about the technical team included:

The technical team don't understand what we need....They're the ivory tower....They never give us what we want.

Stereotyping is rampant, as a functional manager explained.

There really is an us-them mentality. The functional team view the technical team as restrictive, while the latter view the former as unrealistic in terms of budgets, efficiency, volumes.

4.1.4 Losing Autonomy

Functional consultants feel that they have lost autonomy by having to follow the dictates of the technical team and being forced to conform to the language of the CASE tools. In particular, now that systems development work consists primarily of interaction with CASE tools, the functional consultants' designs and code are subject to review by the technical team. This generates much resentment. A functional senior analyst's view is typical.

Without the tech team we would never have made the CASE tools work....But there is some resistance to their presence. When one of them would ask us to change the name of something, we'd resent it. Who are they to tell us what to do?

With the CASE tools mediating systems development activities, many of the tasks previously handled by analysts are now automated. For example, screens and reports are "ergonomically" designed by the tools on receipt of the data items to be displayed and up to 75 percent of the program code for standard, transaction processing systems is generated by the tools on receipt of a few customizing parameters. Functional consultants no longer engage in many detailed design and implementation processes. They resent being excluded from decisions which directly impact the form and functioning of the application systems they are constructing for clients. In particular, the functional consultants object to the technical team developing and installing technical architecture and CASE tools without consulting them, taking exception to having to use technology they did not help design. While some of this resentment may have existed before the onset of tools with the systems software that functional consultants used, this software was typically perceived as "in the background," and hence as dissociated from application work. CASE tools, however, directly confront the functional consultants in the performance of their work, being very much in the "foreground" of systems development work, mediating almost everything the functional team does. A functional consultant commented on the resentment that he and his peers feel towards what they perceive as elitism among the technical consultants.

Tools have developed a tech team of egotists, who feel they have all the knowledge, and that the more they can keep hidden from us, the more knowledge and power they have and can keep.

4.1.5 Asserting Control

The loss of autonomy that the functional analysts feel relative to the technical team frustrates their ambitions to get the job done. The frustration and tension can sometimes lead to rebellious action, to a defiance of Beta's norms that require team play, cooperation, and conformity. A number of functional consultants described resorting to sabotage of the CASE tools in order to reassert their sense of control. For example, on a particularly technically complex project with many CASE tools rigidly enforced by the technical team, the control over systems development work was so effective that, as one functional manager stated,

[It] drove people on the functional team to break the tools right and left. As soon as things started going wrong with the tools, we circumvented them so that we could get on with our work.

A functional consultant recounted a similar tale on another project. His story is worth citing in full.

On this project, the tech team had set up two kinds of ids to use the system. The one was a technical or powerful id which allowed you to romp around in the operating system, create files, etc. The other was an id for the functional people, for the coders, which only gave you access to the coding panel (a menu of options to use the CASE tools) where you could only edit or browse files, compile programs, print program listings, and you could not test some programs, look at some files, or create new files. So we were very restricted functionally, and we found this extremely limiting to our work. But the tools were often causing a lot of unnecessary work. For example, they would time and date stamp the object modules when they were compiled and the test data when they were generated. And these time and date stamps were often out of sync. And if they were out of sync the tools wouldn't let you do anything like use the data to run an object module, so we would have to run to the tech team each time so they could fix things up. The tech team was reluctant to give us technical ids or their passwords so we could use their ids and fix up things ourselves. They thought we'd wreak havoc. But somehow technical passwords were gotten hold of. Some people looked over the tech consultants' shoulders; I had a friend on the tech team so I could get hold of his. And we used these powerful ids to go into the system and we changed our own ids to give us more power, so we had greater functional capability. And we could get on with our work. Of course when all this came out, a big political stink blew up. We were told we weren't team players. But eventually we convinced the tech team that we needed to mani-

pulate a little more of the files. In the end they created a three-tiered system, with those who were fully technical like the tech team having all access, some who were semi-technical like us, who did functional work but needed to sometimes go around the tools, and then the fully functional types like coders who only did application work and only used the coding panels.

So a partial victory had been won for the functional consultants who forced the technical team to give them more computer capability, but the access given was very much on the technical team's terms. The technical consultants managed to regain the control that had been temporarily usurped. They did not give the functional team more powerful ids, which would have given the functional consultants control over when and how to use or bypass the tools. Instead the technical team modified the functional consultants' coding panels so that for a few simple functions the functional consultants could exit the tools to do a few restricted actions outside the realm of the tools. However, this only allowed them to do a little more than they were able to do before their revolt. More importantly it did not give them the option to choose when and how to use the tools.

Rebellions such as these are not common within Beta. The strong corporate culture and ideology of teamwork discourages dissension among the functional consultants, who are concerned with personal career advancement and do not want to be labeled as "troublemakers." However, when rebellious action by functional consultants does occur, it is particularly revealing. On the one hand, it indicates how frustrating task restrictions can be to people with the appropriate expertise who realize that things could be done differently. On the other hand, it demonstrates that, socialization, corporate culture, and career ambitions notwithstanding, individuals can and do act in ways that undermine mechanisms of organizational control.

4.2 Technical Subculture

4.2.1 Technical Perspective

Technical consultants are typically attracted to Beta because of their dual interest in technology and business. For their first few years in the firm, however, business interests play a secondary role, as the specialization of roles on projects demands that they focus on technology. Most technical consultants are more than happy to oblige. While they acknowledge that the purpose of projects is to solve functional problems, they welcome the opportunity to concentrate exclusively on the computer medium and leave the business details to their functional counterparts. Many see the technology as the means through which they can express and display their creativity and a way for them to learn new technical skills which are highly prized

(among their peers and in the larger "hacker" occupational culture to which they feel some affiliation). In this sense, exploiting the technology becomes an end in itself. Technical consultants are **process oriented**, intent on action rather than results. Information technology is more than just instrumentally important to the technical consultants in the execution of their tasks. It carries motivational significance as well. Technical consultants are less focused on the immediate client problem and more interested in finding a unique and elegant solution to the technical problems at hand. While their time orientation is somewhat in the present, it also tends to look beyond the current project to a more abstract, timeless level where technical architectures and CASE tools are perfectible according to some absolute criteria. Technical consultants commonly rationalize why they spend inordinate amounts of time recreating a routine or macro by claiming that their products will be reused on future Beta projects in other clients' sites. Hence (they argue) their work transcends present-time and client-specific boundaries. That such reuse is not commonly realized in practice seems of little consequence to their motivation.

The context of technical consultants' work is defined by the particular hardware and software configuration of the current client's installation. While this context is specific to projects, it is stripped of social or functional content by the technical consultants' exclusive emphasis on the technology. Their attitude towards information technology reflects their greater involvement in process: the technology (both CASE tools and the information technology used to build application systems) is created and manipulated, and hence is not objectified. It is not perceived as a tool for getting their work done, but rather as constituting the arena on which their work is played out.

Akin to their functional counterparts, technical consultants do not see themselves as system developers. While they do subscribe to the "consultant" image, they augment this with a constant reference to their status as technical innovators. This self-image allows them to differentiate themselves from the other team members. One technical senior analyst noted,

The contribution of technical people to Beta is that we are crusaders. We find out all the neat things done in the labs and we figure out how to import these into our monolithic development environment and procedures.

An aspect of Beta that helps to keep the technical analysts motivated is their participation in a strong and active technical community. Part of this community is tied to the computer "hacker" culture beyond Beta (Turkle 1984) with which the technical consultants identify. Technical consultants pride themselves on being creative, which explains their dislike for the functional consultants' preoccupation with getting the application "out the door." Technical consultants are heavily involved with the technology and

they can often avoid interacting with their colleagues and users if they wish. A technical senior analyst remarked:

I prefer to just do the technical stuff. I get quicker feedback that way. You do something and then when you're finished it goes away. You don't have users coming back all the time with changes.

There is a strong identification with being special and different from the rest of the consultants in Beta. One technical manager remarked:

All the technical people are so arrogant. We are the spoilt rotten brats in Beta. We're told we are so wonderful and we tend to believe it. We think functional work is so easy anyone can do it.

One means through which technical consultants reinforce their difference is by maintaining their own internal communication system. One of the local technical groups publishes a regular newsletter, *The Technical Times*, which is distributed to all technical consultants in Beta and which contains reprints of topical articles,⁸ transcripts of presentations given at Beta meetings, and some notes by senior technical managers. Another way of sustaining the technical culture is through monthly local technical group meetings, which usually comprise presentations of the latest technology, research and design innovations so as to keep the technical consultants technologically stimulated. One such meeting, for example, spanned a whole day and was devoted to the latest user interface technologies, with a number of vendors bringing in their products and giving hands-on demonstrations. Interspersed among the demonstrations were presentations by guest speakers and in-house personnel on issues in ergonomic design and the dimensions of the person-machine interface. Such meetings and communications serve to strengthen the shared meanings, values, and norms of the technical subculture, as well as providing a forum in which to transmit and reinforce such shared meanings and interests. The result is a reaffirmation of technical consultants as special and different from the rest of the consultants in the organization. This subculture identity in turn helps to sustain the polarization of consultants on the project teams.

4.2.2 Discounting the Functional Consultants

The technical consultants tend to stereotype other team members (whether functional consultants or client data processing personnel) as "tool users" and hence as having little understanding of the information technology underlying the tools. This allows the technical consultants to rationalize their need to hide the tools from these "user types." On projects, the status of actors in Beta and client organizational hierarchies is less important than actors' ability to wield technical control. On one project, for example, a junior technical consultant, who had been with

Beta for barely a year, was able to limit the discretion of functional consultants (two and three years his senior) and client data processing personnel (with five to eight years technical experience) through his authorization of their CASE tool user profiles. His attitude is evident in this rationale he gave for restricting the computer capability of other team members.

You show them [the non-technical members of the team] the functions of the tool gradually, not all at once, as you want to get them going as fast as possible. So you don't show them things they won't need, or things that are too complicated as you don't want to confuse them. So the principle is show them as little as possible.

In other circumstances, this technical consultant's approach might be taken as evidence of a careful teacher who does not want to overwhelm his/her students. However, on this particular project, all the functional team members and client data processing personnel were technically experienced. It seems that the technical team's world-view engenders a perception of other team members as necessarily incompetent without tools. The consultant's attitude is typical of most technical consultants, whatever the reality of their functional team's experience and knowledge.

This dim view of their functional contemporaries is, however, becoming a self-fulfilling prophecy. The functional consultants' lack of exposure to technical issues coupled with Beta providing little technical training to new recruits is creating a growing pool of technically illiterate functional consultants. This has the desired effect of increasing the use of CASE tools on projects (hence increasing Beta's project leveraging factor) while reinforcing the functional team's dependence on the technical team. It also has an unanticipated consequence, as it requires technical consultants to be responsive to the now more-dependent functional consultants. Technical consultants find themselves performing technical support tasks (repairing databases, installing new versions of tools, creating backup copies of software and data, training tool users, and answering technical questions) when they would prefer to be building advanced and complex technical routines. A technical manager remarked:

With the tools and technical architecture we completely hide CICS and DB2 from the programmers, and can get incredible productivity from them. But we also get uneducated programmers, who can't handle the smallest technical problem, and at the first sign of trouble they go screaming for the tech team.

4.2.3 Resisting the Results Orientation

Notwithstanding the extensive control that technical consultants have over the conditions under which the

functional team operates, the inherent results orientation of the Beta culture causes much frustration among technical consultants. For technical consultants, Beta's results orientation is a major restriction on their creativity and a constraint on their ability to refine the technical infrastructure they construct and maintain. They feel that a results orientation is myopic and that such a short-term strategy inhibits the development of good technical solutions. Given the dependence of the whole team on the CASE tools, the technical team's mandate is to get the tools operational on the current computer environment as soon as possible, in any way possible. But this does not allow sufficient time for technical consultants to develop the smart, elegant solutions they would like to. A technical consultants commented that

The budgetary restrictions on the project cause problems in scope; they force a narrow view on the tech team. This is frustrating for us, the technical team, as we see and know what should be done to improve, refine, and generalize the tools we carry around from project to project, but we can't do that....So it is frustrating for the technical types who may have great ideas, but who don't have the time to develop or implement them.

It is interesting to look at this issue of results orientation from the other side. A functional senior analyst felt that a belief in the "technical fix" pervaded Beta's technical consultants.

There is a preoccupation with tools in Beta, particularly from the technical groups. They are always in search of the golden goose that will save time and money. But that's just too simple.

Another functional consultant remarked on technical team priorities.

There is a feeling that they [the technical consultants] do not want to release stuff until it is perfect....But right now we need basic transportation. And we would rather they give us something to walk with, and then they can enhance it later to give us a racing car.

The standoff between these two orientations reflects the functional interest in results and the technical interest in technological innovation. It is a conflict that has to be managed constantly and the burden of this falls on the shoulders of the project managers, who must support their functional analysts in getting the project completed on time and within budget, but must also keep their technical analysts motivated to provide a reliable, efficient and useful system infrastructure.

4.2.4 Summary of CASE Tools Research Study

The deployment of CASE tools in Beta triggered structural changes within the project teams, which institutionalized the existing, formalized fragmentation of expertise into technical and functional groupings. Such a duality of interests, orientations, skills, and tasks on projects undermines the homogeneity of the Beta "team" ideology by breeding subcultures and territorialism. It results in tension and conflict on project teams that sometimes lead to eruptions. The functional consultants seem typically to lose out in these confrontations, as CASE tools are the stated policy of Beta, and the technical consultants, as implementors of this stated policy, have legitimacy and Beta's resources on their side. Rebellion by functional consultants is a way for them to reassert the autonomy they feel the technical team has usurped. In time, such resentment may diminish as new functional consultants enter Beta and take the presence of a powerful technical team for granted. Not being aware of the prior division of labor, and not having been exposed to projects where technical consultants were the "backroom guys," these functional consultants are unlikely to feel a loss of control, centrality, and territory.

The following section re-examines the findings described above by interpreting them in terms of a broader social theoretic framework so as to derive more general insight into relations between technical experts and functional workers around the deployment of information technology.

5. THEORETICAL INTERPRETATION

The introduction of CASE tools in information systems development can be understood as an instance of the more general phenomenon increasingly pervading contemporary organizations: the deployment of information technology in core production activities. In the particular instance of CASE tools in Beta, the production work being mediated by information technology is the development of information systems and the particular information technology being deployed is a set of capabilities that support/automate the activities of systems development. Recognizing this, the specific findings about social relations on project teams using CASE tools can be articulated in terms of the more generic organizational processes of which they are a microcosm.

The insight provided by this study pertains to changes in the division of labor and the relations of dependency on project teams following the introduction of CASE tools. This suggests that any substantial mediation of production work by information technology can be expected to lead to changes in the division of labor and patterns of dependency among the actors involved. Introducing information technology commits a production process to a technically complex infrastructure of hardware, software, and procedures that necessarily involves new forms of expertise. As

a result, specialized skills are needed to ensure the reliable and effective mediation of production by information technology. At least two ways of providing these skills are possible.

1. If the existing production/functional personnel are able to adequately manipulate the information technology, they will be required to integrate technical skills into their work. In this case, functional workers also become technical experts and social relations among workers need not be disrupted (although disruptions may result from the role conflict, ambiguity, and overload that such an infusion of new tasks and skills could engender within individuals). While this may appear to be upskilling of work, in that the workers now acquire technical skills, it need not be. Job upskilling only occurs if the new skills increase the discretion and autonomy of the workers on their jobs. If the skills are required just so that workers can perform the same tasks they executed before (with the same level of autonomy), there has been no upskilling of work. It is important to note that the relationship posited here refers to work and not to workers. The processes of deskilling affecting jobs are not synonymous with those affecting individuals (Attewell 1985; Lee 1981). For example, while a particular task may be deskilled by being made simpler and more routine, the particular job incumbents may not be deskilled for they may now work at the more skilled aspects of the job, with the deskilled task being executed by a different set of workers (who may well find that their jobs have been upskilled).
2. Given the extreme specialization of roles commonly experienced in organizations, however, it is unlikely that functional workers will have the capability, resources, or inclination to develop and maintain the information technology that facilitates their daily work, end-user computing notwithstanding. As the demand for and supply of sophisticated and complex computer capability escalates in advanced industrial economies, it is probable that the information technology deployed in production work will be beyond the scope of the average end-user. In this case, functional workers will merely use the information technology, while outside expertise in the form of technical specialists will be imported into the production process to develop and maintain it. New technical expertise and new technology-management tasks are introduced into a given production process and the existing division of labor and patterns of dependency will certainly be affected, as evidenced in Beta's experiences.

In Beta's case, changes in tasks and expertise are reflected through changes in social relations among project team members. Rephrasing in more general terms, social relations among the key actors in the production arena will be affected as a consequence of deploying information technology in production. Power is a feature of all forms

of social relations (Giddens 1976, p.112), so that one of the ways in which the nature of change in social relations can be investigated is through studying shifts in power relations among key actors. Power is the means of getting things done and, as such, actors mobilize different resources that they bring to bear on their social interaction. It is through differential possession of and access to these resources that power is exercised. In the light of the findings presented above, it is expected that in the context of production work mediated by information technology, technical expertise and unrestricted access to information technology become significant resources around which power is mobilized. The study of Beta further demonstrates that CASE tools, as central elements in social interaction, generate conflict between functional and technical consultants on the project teams. Given differences among technical and functional workers in perceptions, interests, and work-pressures, the differential distribution of technical resources can be expected to lead to or accentuate social conflicts around any production process being mediated by information technology.

By deploying information technology in production, traditional bases of expertise and authority (hence power) in the existing production process are threatened. With the increased dependence on technical expertise, conflict arises between the newly-arrived technical experts and the established functional workers whose expertise and authority is challenged and changed through the mediation of production work by information technology. How this conflict is played out across various production arenas remains open to empirical elaboration. The technical experts may triumph, institutionalizing their technocratic dominance, or production workers may reassert control through their continued resistance to the technical dependence that is now inherent in their computer-mediated production tasks. Different outcomes will be generated across different contexts and different outcomes may be generated over time within the same context. While such outcomes can never be predicted unequivocally, we can determine the likelihood of different patterns of response based on an understanding of contexts, actors, and resources. For example, it is expected that where production workers have established legitimacy and influence -- say they are accredited professionals (physicians or lawyers) -- it is more likely that institutionalized bases of authority will persist, even in the face of information technology. On the other hand, where production workers have little credibility beyond the confines of their employing organization -- as in the case of Beta's functional consultants -- the infringement of authority by technical specialists is more probable.

In Beta, there was a shift of power to technical experts as prior, more established functional bases of power were undermined. While the technical consultants exercised considerable control over the circumstances under which functional consultants worked, their power was not inviolate. Functional consultants, as knowledgeable and

capable agents, occasionally were able to recognize the constraints on their behavior and take action to subvert the power of the technical consultants. Thus, by refusing to conform to the tools and the "team" way of doing things, such functional workers reasserted their agency, underscoring the notion that power is always relational -- not a static property of an individual or institution -- and not only realized through social interaction (Giddens 1984). They also risked their status and job security. The message here is that, when provoked sufficiently, individuals can and do rebel against structural and technological imperatives.

The frequency with which "power struggles" occur, and the extent to which functional workers resist the conditions of the power asymmetry in their work context, will determine the amount of dominance technical experts amass. Where such resistance is weak or sporadic, the differential distribution of technical resources will be reproduced over time. Where technical experts are able to generate outcomes by affecting the conduct of functional workers -- through the application of technical knowledge or manipulation of information technology -- such reproduction will occur. Through their action or lack thereof, technical and functional actors thus recreate the conditions of their dependence relationship. In time, such power asymmetries become institutionalized into structures of domination. However -- and this point must be stressed -- such patterns of domination are not deterministic and where actors deliberately take action that challenges existing conditions and resource distributions, they have the potential to transform rather than reproduce the power asymmetry. While dissension was present at Beta, it was not sufficiently strong or organized enough to achieve transformation and, as was seen, the technical experts' dominance was sustained (at least for the time period examined by the current study). Notwithstanding this, it is possible that in the future within Beta, as CASE tools become more general, reliable and simple to use, the functional workers will lose their reliance on technical experts. This would attenuate the power of the technical experts as the value of resources will have eroded. The existing patterns of domination will in such circumstances not be reproduced and new social structures will emerge.

6. CONCLUSIONS

This paper has sketched out the ways in which social relations on project teams can be disrupted and changed as a consequence of deploying CASE tools. The findings suggest implications for practice as well as directions for future research. Practitioners introducing CASE tools into systems development efforts need to recognize that such implementation will affect social relations among project team members, reflecting the changes that result when any significant mediation of work by information technology is realized. Being sensitive to the potential disruption is an important first step in managing the changes that CASE tools will inevitably bring to the development process.

More research is needed to determine the conditions under which tensions between functional and technical project personnel may be mitigated and when they are exacerbated. Of particular interest is determining how systems development tasks are distributed among the three sets of agents: application developers, tool facilitators, and the actual CASE tools. Little is known about the longevity of the disruptions and whether the territorialism, resentment, and rebellion are transitory or more deeply rooted disjunctures. There is some preliminary evidence for a generational effect, which leads to the speculation that as CASE tools become more fully part of the systems development "landscape," they will become more institutionalized, and systems developers will be socialized into taking them for granted. As such, tools may lose their potency as overt symbols of power and harbingers of territorial threat. Less conflict may result.

Changing the social relations around the systems development process, while clearly affecting the actors directly involved, will also likely affect the indirect actors -- the users -- who ultimately are the consumers of the final product. CASE tools are often touted as increasing user participation. How use of tools changes social relations between users and system developers, and across different levels of users, contexts, and time, are significant areas of future research. This study examined the use of CASE tools in a software consulting firm, but CASE tools are also being implemented with increasing rapidity in traditional data processing environments. While relations between tool users and tool facilitators in such environments are anticipated to resemble those discussed above, the unique practices of in-house systems development provide a set of dimensions that may alter the social relations affected by CASE tools. How the various shifts in system development tasks, responsibility, and authority -- occasioned by CASE tools -- are interpreted and acted on across different organizational contexts will require comparative investigations over time to determine how changes are institutionalized.

As production processes become mediated by information technology, new technical expertise is introduced into the production work. An important issue arising from this, not explored in the current discussion, concerns the management of the new expertise. Given that production work is central to the operation of an organizational unit, mediation of this work by information technology means that the expertise responsible for maintaining the integrity of that mediation is critical to the ongoing functioning of the unit. A key question here is how does the existing authority structure integrate the new forms of expertise without undermining its power?

In general, the social relations view used here to interpret the automation of systems development processes promises to be a useful perspective within which to understand the interdependent and dynamic social changes engendered by the deployment of information technology in organizations.

7. REFERENCES

- Agar, M. H. *The Professional Stranger*, New York: Academic Press, 1980.
- Attewell, P. "The De-Skilling Controversy." Working Paper, Sociology Department, State University of New York, Stony Brook, New York, 1985.
- Boland, R. J. "Control, Causality and Information Systems Requirements." *Accounting, Organizations and Society*, Volume 4, Number 4, 1979, pp. 259-272.
- Carlyle, R. E. "Where Methodology Falls Short?" *Data-mation*, Volume 34, December 1988, pp. 179-191.
- Case, A. F. "Computer-Aided Software Engineering: Technology for Improving Software Development Productivity." *DataBase*, Fall 1985, pp. 35-43.
- Freedman, D. H. "Programming without Tears." *High Technology*, April 1986, pp. 38-45.
- Giddens, A. *The Constitution of Society: Outline of the Theory of Structure*, Berkeley, California: University of California Press, 1984.
- Giddens, A. *New Rules of Sociological Method*, New York: Basic Books Inc., 1976.
- Kling, R., and Scacchi, W. "The Web of Computing: Computer Technology as Social Organization." *Advances in Computers*, Volume 21, 1982, pp. 1-90.
- Lee, D. J. "Skill, Craft and Class: A Theoretical Critique and a Critical Case." *Sociology*, February 1981, pp. 56-78.
- Markus, M. L. "Power, Politics, and MIS Implementation." *Communications of the ACM*, Volume 26, 1983, pp. 430-444.
- Orlikowski, W. J. *Information Technology and Post-Industrial Organizations: An Exploration of the Computer-Mediation of Production Work*, Unpublished Doctoral Dissertation, Stern School of Business, New York University, 1988.
- Riley, P. "A Structurationist Account of Political Culture." *Administrative Science Quarterly*, Volume 28, 1983, pp. 347-414.
- Schein, E. H. *Organizational Culture and Leadership*. San Francisco: Jossey-Bass, 1987.
- Stamps, D. "Cranking Out Productivity." *Datamation*, July 1, 1987, pp. 55-58.
- Turkle, S. *The Second Self: Computers and the Human Spirit*, New York: Simon & Schuster, 1984.
- Van Maanen, J. "The Fact of Fiction in Organizational Ethnography." *Administrative Science Quarterly*, Volume 24, 1979, pp. 539-550.
- Van Maanen, J. *Tales from the Field*, Chicago: University of Chicago Press, 1988.
- Weick, K. E. "Theoretical Assumptions and Research Methodology Selection." In W. F. McFarlan, Editor, *The Information Systems Research Challenge*, Boston Massachusetts: Harvard Business School Press, 1984, pp. 111-132.

ENDNOTES

1. It is estimated that, in the U.S. to date, only seven or eight percent of information system workers have been exposed to CASE tools (Carlyle 1988).
2. Beta has a single career path with the stages junior analyst (two years), senior analyst (two years), junior manager (4 years), project manager (2 to 3 years), and senior project manager.
3. **Functional expertise** refers to knowledge of an industry, such as aerospace or banking, and knowledge of functional areas, such as production, marketing or sales. **Technical expertise** refers to knowledge of software products, such as operating systems or database management systems, and familiarity with various hardware platforms.
4. Development inefficiencies were due to misinterpretation, ambiguity, rework, and multiple, separate learning curves.
5. The technical architecture is that set of software modules common to most programs of an application system and which is typically technically complex (e.g., screen mapping, screen communication, database I/O, macros).
6. CASE tools in Beta emerged out of individual project endeavors and are not general enough to be fully portable across multiple environments. As each of Beta's clients has a unique hardware and software environment, the CASE tools have been adapted at each client installation to fit the particular computer configuration at hand.
7. Comments in italics constitute statements made by Beta personnel during the field study.
8. A recent issue included, for example, the articles "The Rise of Techno-Nationalism" by Robert Reich reprinted from *The Atlantic* and "No Silver Bullet" by Fred Brooks Jr. reprinted from *IEEE Computer*.